

课程设计报告

--基于Flask框架和微信开发语言实现的微信点名系统

加兴华 18300290007

陈浩贤 18307110276

于淼芃 19307130203

0 小组分工

1 题目介绍和需求分析

1.1 题目介绍

- 1) 题目背景
- 2) 涉及信息
- 3) 基本功能

1.2 需求分析

1.3 数据流图

1.4 数据字典

2 系统设计

2.1 功能规划

2.2 算法设计

- 2.2.1 登录算法
- 2.2.2 课程加载算法
- 2.2.3 课程考勤历史查看算法
- 2.2.4 老师发起签到算法
- 2.2.5 学生签到算法
- 2.2.6 缺勤查询与修改算法
- 2.2.7 统计算法

2.3 微信小程序系统设计

- 2.3.1 系统全局变量设计
- 2.3.2 系统文件页面对应设计

2.4 后端设计

2.5 数据库设计

- 2.5.1 实体和联系
- 2.5.2 E-R图
- 2.5.3 关系模式
- 2.5.4 数据库表的建立
- 2.5.5 物理引擎的选择
- 2.5.6 索引建立
- 2.5.7 触发器建立

3 功能介绍

3.1 教师端

- (1) 首次登录绑定
- (2) 发起签到任务
- (3) 查看签到记录
- (4) 下载考勤记录
- (5) 缺勤记录修改
- (6) 查看统计信息
- (7) 联系开发人员

3.2 学生端

- (1) 首次登录绑定
- (2) 进行签到任务
- (3) 查看签到记录
- (4) 查看统计信息
- (5) 联系开发人员

4 开发平台及框架

Flask

微信开发者工具

MySQL

5 系统安装部署说明

附：系统使用说明

1. 进入小程序
2. 教师登录与相关功能
3. 学生登录与相关功能
4. 用户反馈

0 小组分工

陈浩贤--数据库及python后端设计、数据录入工作、期中汇报录制、期末报告撰写

于淼芃--微信小程序前端设计、数据录入工作、期中汇报录制、期末报告撰写

加兴华--小程序与后端交互算法设计、期中汇报ppt及讲稿撰写、期末报告撰写

1 题目介绍和需求分析

1.1 题目介绍

1) 题目背景

一门课有多名学生选修，一名学生可以选修多门课程；一位老师可以讲授多门课程，一门课程可以由多名老师进行讲授。

2) 涉及信息

学生的基本信息：姓名、学号、院系、专业、性别、年级等

3) 基本功能

1.出勤统计：缺勤次数统计、查询考勤情况、请假情况；能从不同维度进行统计分析

2.点名系统能进行后台设置：扫码有效时间、签到地址、批量导出出勤数据

3.通过扫描二维码，进行考勤管理

1.2 需求分析

根据选题，本产品旨在实现一款用于老师考勤学生签到的微信小程序，为确保使用者满意，我们首先进行了需求分析。

本设计产品的主要用户为老师和学生，老师需求包含发起一次签到并设置考勤条件，能够为因特殊情况导致无法签到的学生设置请假或出勤状态，能够查询自己课程的学生出勤情况并下载考勤记录；学生需求包含进行一次签到，以及查询自己某一门课的具体考勤情况。

此外，所有用户（包括老师和学生）往往还会有进行数据分析的需求，针对某一门课程的各种维度进行数据统计。

本设计产品的安全性需求主要有两方面：一是要保证学生的签到是合乎考勤条件的，即本人在规定时间于现场才能签到，安全性体现为学生不能通过代签等手段虚假签到，这方面需要进行用户身份鉴别；二是要而是要保证数据库中重要或敏感的数据不会被非法读写，这一方面需要数据库的存取控制，包括定义用户权限等。

为了满足这部分的需求，我们的系统会录入用户进入小程序时产生的openID。openID是一种识别用户身份的ID，同一用户在同一微信小程序上的openID总是相同且不变的，并且不同用户的openID均不同。由于其具有每台设备的唯一性，因此用作防止登录他人账号是很合适的；此外，我们的签到操作在检查时间和地点信息正确后才对学生开放，满足了第二个方面的安全性需求。

1.3 数据流图

根据上述需求分析，我们作出以下数据流图：

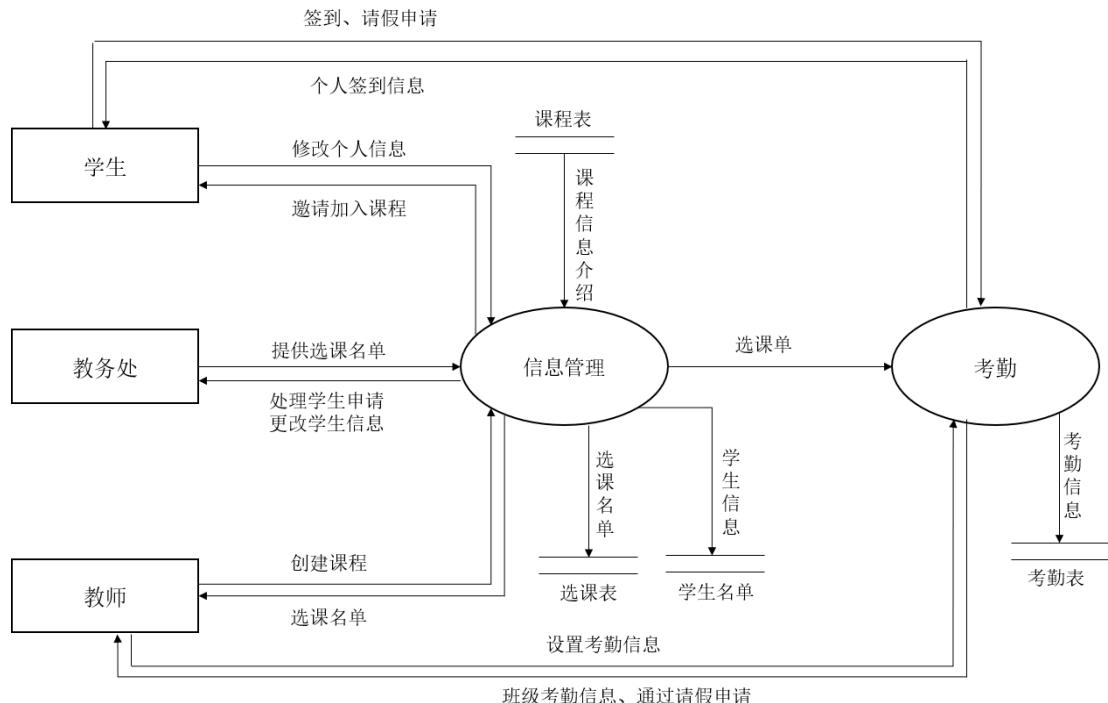


图1-1 数据流图

1.4 数据字典

接下来，我们以复旦大学的实际情况作为参考，可以列出数据字典。以下为部分的数据字典：

数据名称：记录号 说明：标识每次考勤记录 类型：INT(10) 长度：10 取值范围：0000000001~9999999999	数据名称：学号 说明：标识每个学生，以复旦学号为例 类型：VARCHAR(11) 长度：11 取值范围：0000000001~9999999999	数据名称：性别 说明：学生性别 类型：VARCHAR(6) 长度：6 取值范围：'male', 'female', 'unknown', 'others'
--	---	---

数据名称：出勤统计次数 说明：记录该课程进行考勤的次数 类型：INT(4) 长度：4 取值范围：默认取值为0，取值0~16	数据名称：年级 说明：学生年级 类型：INT(4) 长度：4 取值范围：2010~2025	数据名称：课程代码 说明：标识每个课程 类型：VARCHAR(20) 长度：20 取值范围：字母和数字共同构成
---	---	---

数据存储名：考勤信息表 说明：记录考勤的信息 数据流来源：学生 数据流去向：考勤系统 数据组成：记录号+学号+课程代码+记录时间+出勤情况 数据量：5000条/天
--

图1-2 数据字典

2 系统设计

2.1 功能规划

清楚上述需求后，我们为产品功能做了以下规划：

- 主体功能：老师发起签到和学生进行签到
- 必要功能：老师修改缺勤学生考勤状态为请假或出勤
- 附加功能：老师可以查看自己课程出勤情况、学生可以查看自己在某一门课的出勤信息，以及对所有用户开放维度查询
- 内在功能：将账户信息与手机绑定防止代签，同时能核验地点、时间信息来限制学生的签到操作

2.2 算法设计

2.2.1 登录算法

这一部分旨在验证本人操作从而杜绝代签现象，实现的方法为进入登录界面时获取设备的openId，在用户提交姓名学号信息时一同传递给后端，如果是首次登录则将openId和其姓名、学号（工号）进行绑定；如果已经登陆过则会将该次的openId和用户信息表中的进行比对，如果不同则拒绝登录请求。

在通过登录请求后，将后端返回的学院、ID、年级、专业信息存至js中的全局变量，因为后续很多的查询及展示都涉及这些信息，所以在使用时有必要一直将它们保存在缓存中，通过空间换取运行速度，这样无需再通过后端从数据库中查询这些信息。

2.2.2 课程加载算法

登录成功后，页面应当自动跳转到用户信息界面，以待用户切换到课程或是统计界面。为了保证用户后续切换不用等待信息加载，在用户信息界面就应当已经拿到课程信息，所以在本界面前端就向后端发送请求，传入之前获得的id信息，并返回该学生或老师的课程信息表。因为一个学生或老师拥有的课程是稳定的，无需重复查询，所以这里也将返回的课程信息表置为全局变量来便于后续操作。以下为该算法的具体实现代码。

```
1 wx.request({
2     url: app.globalData.path + '/login',
3         //传递WechatID, 输入的ID, 姓名
4     data: {
5         WechatId: app.globalData.openID,
6         id: ID,
7         role: app.globalData.identity,
8         name: userName
9     },
10    header: {
11        'content-type': 'application/json'
12    },
13    method: "POST",
14    success: function(res) {
15        switch (flag) {
16            //出现错误，后端返回0, 前段返回登录失败字样
17            case 0:
```

```
18     wx.showToast({
19         icon: 'error',
20         title: '登录失败',
21         duration: 1000
22     })
23     break;
24 //登录成功，后端返回1
25 case 1:
26     app.globalData.isLogin = 1;
27     app.globalData.isSubmit = true;
28     app.globalData.userName = e.detail.value.userName;
29     app.globalData.ID = e.detail.value.ID;
30     //身份为学生时的操作
31     if(app.globalData.identity==0){
32         console.log(app.globalData.bothid)
33         //请求课程列表
34         wx.request({
35             url: app.globalData.path+'/student/search_lesson',
36             data: {
37                 Id: app.globalData.bothid
38             },
39             header: {
40                 'content-type': 'application/json' // 默认值
41             },
42             method: "POST",
43             //传递课程列表
44             success:function(res){
45                 var json=res.data
46                 app.globalData.lessonList = json
47             }
48         })
49     }
50     //身份为老师时的操作
51 else{
52     //请求课程列表
53     wx.request({
54         url: app.globalData.path+'/teacher/search_lesson',
55         data: {
56             Id: app.globalData.bothid
57         },
58         header: {
59             'content-type': 'application/json' // 默认值
60         },
61         method: "POST",
62         //传递课程列表
63         success:function(res){
64             var json=res.data
65             app.globalData.lessonList = json
66         }
67     })
68 }
69 //前段返回登录成功字样
70 wx.showToast({
71     icon: 'success',
72     title: '登录成功',
73     duration: 1000
74 })
75 setTimeout(() => {
```

```

76             wx.navigateBack()
77         }, 1000);
78     break;
79 }
80 }
81 })

```

代码2-1 登录与课程加载js代码

```

1 @app.route('/login', methods=['POST']) # 登录
2 def login():
3     if request.method == 'POST':
4         if not request.json:
5             return jsonify('not json')
6         else:
7             data = request.get_json()
8             return_data = dict()
9             wechatid = data['wechatId'] # 获取传入的参数
10            id = data['id']
11            role = data['role']
12            name = data['name']
13            if role == 1: # 用户为老师
14                db_data = search_teacherwechatid, id, name) # 在数据库中查找
15                if db_data: # 查询有结果
16                    return_data['Id'] = db_data['Id']
17                    return_data['Name'] = db_data['Name']
18                    return_data['flag'] = 1
19                else: # 查询无结果
20                    return_data['flag'] = 0
21            if role == 0: # 用户为学生
22                db_data = search_studentwechatid, id, name) #在数据库中查找
23                if db_data: # 查询有结果
24                    return_data['Id'] = db_data['Id']
25                    return_data['Name'] = db_data['Name']
26                    return_data['Dep'] = db_data['Dep']
27                    return_data['Major'] = db_data['Major']
28                    return_data['Sex'] = db_data['Sex']
29                    return_data['Grade'] = db_data['Grade']
30                    return_data['flag'] = 1
31                else: # 查询无结果
32                    return_data['flag'] = 0
33            return jsonify(return_data) # 返回json数据
34        else:
35            return jsonify('not POST')
36
37
38 def search_studentwechatid, id, name):
39     result = studentinfo.query.filter_by(Id=id).first()
40     # 对应的sql语句为: select * from studentinfo where id='id', 'id'表示传入参数
41     id
42     if result is None: # 查询无结果, 返回None
43         return None
44     result = result.to_dict() # 将查询结果转化为字典
45     if result['Name'] != name: # 检查名字是否对应
46         return None
47     if result['WeChatId'] == wechatid: # 检查openID是否对应
48         return result

```

```

48     elif result['WeChatId'] is None: # 如果该用户未注册，则对表进行修改，加入对应的openID
49         try:
50             data = db.session.query(studentinfo).filter_by(Id=id).first()
51             data.WeChatId = wechatid
52             db.session.commit()
53             db.session.close()
54             return result
55         except:
56             db.session.rollback()
57             return None
58     else:
59         return None
60
61
62 def search_teacher(wechatid, id, name):
63     result = teacherinfo.query.filter_by(Id=id).first()
64     # 对应的sql语句为: select * from teacherinfo where id='id', 'id'表示传入参数id
65     if result is None: # 查询无结果，返回None
66         return None
67     result = result.to_dict() # 将查询结果转化为字典
68     if result['Name'] != name: # 检查名字是否对应
69         return None
70     if result['WeChatId'] == wechatid: # 检查openID是否对应
71         return result
72     elif result['WeChatId'] is None: # 如果该用户未注册，则对表进行修改，加入对应的openID
73         try:
74             data = db.session.query(teacherinfo).filter_by(Id=id).first()
75             data.WeChatId = wechatid
76             db.session.commit()
77             db.session.close()
78             return result
79         except:
80             db.session.rollback()
81             return None
82     else:
83         return None

```

代码2-2 登录接口python代码

2.2.3 课程考勤历史查看算法

当用户选中一门课并进入其历史签到记录选项，前端应将所选课程代码和用户id传入后端，后端返回通过执行SQL语句的记录数据，数据存于前端的局部变量中用于展示：用户为老师时，展示该门课学生的考勤历史情况；用户为学生时，展示自己的考勤情况。此外我们还在前端对数据做了进一步处理：老师端界面会强调最近一次的考勤情况；学生端则会有一个整体考勤情况的统计。

2.2.4 老师发起签到算法

当老师设好签到条件并发布后，前端会将课程代码、当前时间传入后端，后端自动在考勤表中为每一位选课学生插入“缺勤”记录，同时将考勤表中对应的出勤统计次数加一。同时前端会生成一个含有签到课程、时间、地点信息的二维码，老师通过截图保存，供学生扫描。

2.2.5 学生签到算法

学生进入签到界面后，系统会自动加载之前缓存中的学生个人信息，而后学生需要点击核验地理位置，若通过则会往后台传入课程代码及学生id，后台会将这一天中这门课程该学生的“缺勤”状态变更为“出勤”。

2.2.6 缺勤查询与修改算法

当老师进入该界面，系统首先根据课程代码向后端索要签过到的日期，待老师选中一个后，再次传入该日期和课程代码，传回缺勤学生名单并将其可视化，老师可以选择学生，选定后可以提交请假或出勤，系统会向后端传入所选日期、课程代码、学生id、以及要更改的状态，如果数据库更新成功则返回“success”，反之返回“false”。

2.2.7 统计算法

老师和学生可以进入统计界面，当用户选择好要统计的课程及维度并确认后，系统向后端传入课程代码以及维度，后端及数据库处理后返回相应维度的统计结果并将之以柱状图的形式可视化。

2.3 微信小程序系统设计

2.3.1 系统全局变量设计

正如2.2中所述，在同一名用户登录入系统后，很多信息在从数据库中索取后可以暂存起来，用空间换取查询工作量的减少。经总结，我们所有设计的全局变量如下所示：

全局变量名	变量意义
userInfo	用于获取用户设备等信息
userName	用户登录姓名
institute	专门用于展示用户界面中学院的变量
ID	专门用于展示用户界面中id的变量
isSubmit	是否允许登录的检查标志
grade	学生的年级
major	学生的专业
dep	学生的学院
warn	报错变量
identity	判断用户是老师还是学生的标志
chosenClass_name	课程列表中所选的一个的名称
chosenClass_teacher	课程列表中所选的一个的老师
chosenClass_code	课程列表中所选的一个的代码
chosenClass_num	所选课程的总人数
chosenClass_times	课程列表中所选的一个的签到总次数
openID	当前用户设备独有的设备号
bothid	用于存储师生的id
isLogin	是否登录成功的检查标志
lessonlist	课程列表
path	服务器ip地址

表2-1 全局变量表

2.3.2 系统文件页面对应设计

文件名	对应页面 (导航栏文字)
index	欢迎来到签到小程序 (初始界面)
userInfo	个人信息 (展示个人信息)
editInfo	登录 (进行登录操作)
myClass	课程信息 (列出所参加的课程)
statistics	统计信息 (展示统计信息)
checkIn	签到 (学生: 扫描二维码进行签到及查看历史记录; 老师: 发起签到, 查看历史记录, 查看并修改缺勤情况)
checkIn_stu	签到 (用于学生提交签到)
checkIn_his	历史签到记录 (用于学生查看个人签到记录)
checkIn_his_tea	历史签到记录 (用于教师查看签到情况与记录, 并下载考勤表)
launch1	发起签到任务 (用于教师发起签到任务)
markLeave	缺勤情况查询与修改 (用于教师查看并修改考勤记录)

表2-2 系统文件页面对应表

2.4 后端设计

路径名	对应功能
/wxlogin	获取微信openID
/login	登录
/teacher/search_lesson	查找老师教授的课程
/teacher/launch	老师发布课程，对课程表修改
/teacher/search_leave	查找缺勤信息
/teacher/update_attend	老师更改学生请假状态
/student/search_lesson	查找学生上的课
/search/for_student	学生界面查找个人某课考勤信息
/search/for_teacher_all_lesson	老师界面查找某课程所有考勤信息,输入课程名，输出每次考勤的统计
/search/for_teacher_a_lesson	老师界面查找某课某时间全体学生考勤信息
/search/for_all	统计界面
/create_download	创建考勤记录表格的excel文件
/download/ <filename>	下载对应的文件
/delete/ <filename>	删除对应的文件

表2-3 后端接口对应功能

2.5 数据库设计

2.5.1 实体和联系

通过需求分析，我们可以得到以下实体和联系：

实体：

- 1) 学生。属性：姓名、学号、院系、专业、性别、openID
- 2) 老师。属性：姓名、工号、openID
- 3) 课程。属性：课程名、课程代码、选课人数、出勤统计次数、上课地点

联系：

- 1) 学生与课程之间：选课 多对多
- 2) 学生与课程之间：出勤 多对多 联系的属性：出勤时间、出勤情况
- 3) 老师与课程之间：讲授 多对多

2.5.2 E-R图

根据以上分析的实体信息，可以得到以下的E-R图。

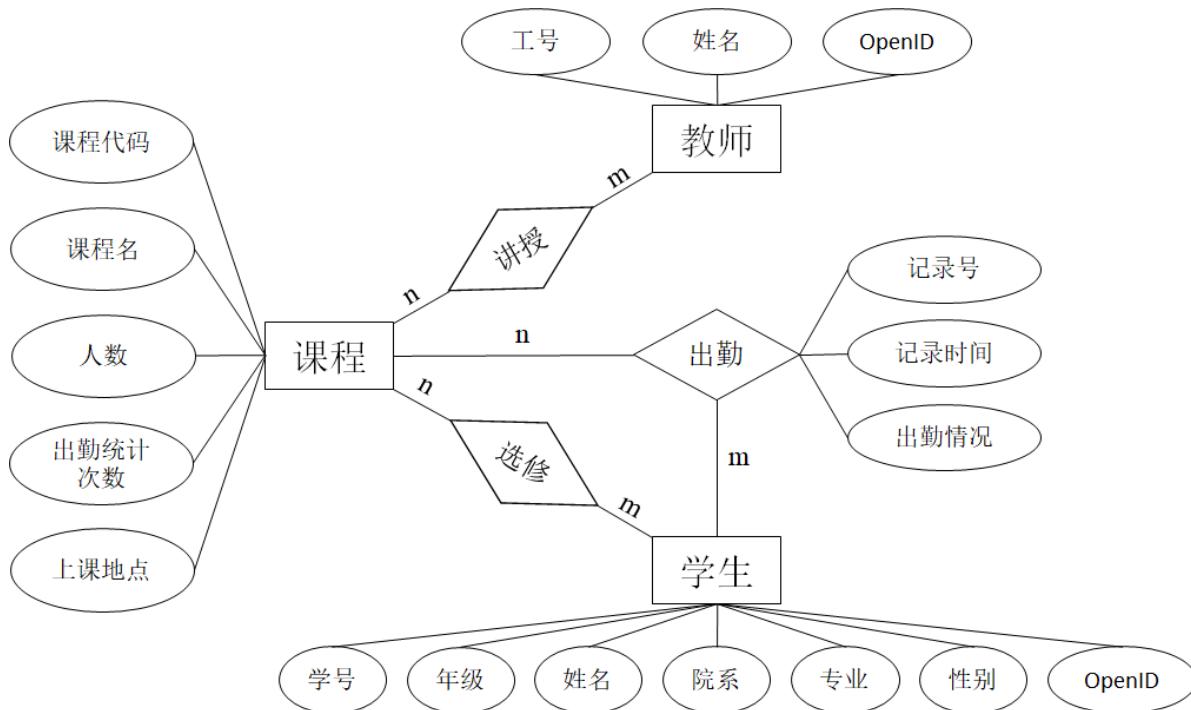


图2-1 E-R图

在E-R图中，我们将上述的实体及其联系绘制出来。相对于以上的有关实体和联系的分析，为了更好地对出勤信息进行管理，我们在出勤这一关系中加入了考勤记录号这一属性。

2.5.3 关系模式

根据E-R图，我们建立了以下的关系模式。其中用下划线标出的为表的主键，蓝色字体为外键。

学生信息表：

(学号, 姓名, 院系, 专业, 性别, 年级, openID)

教师信息表：

(工号, 姓名, openID)

课程信息表：

(课程代码, 课程名, 选课人数, 上课地点, 出勤统计次数)

老师授课表：

(课程代码, 开课老师工号)

选课信息表：

(课程代码, 学生姓名)

考勤信息表：

(记录号, 学生姓名, 课程代码, 记录时间, 出勤情况)

根据以上的关系模式，我们建立如下的模型图。在该模型图中，我们标出了各表对应字段的名称、数据类型、以及主键和外键，同时也把表与表之间的关系表示了出来。

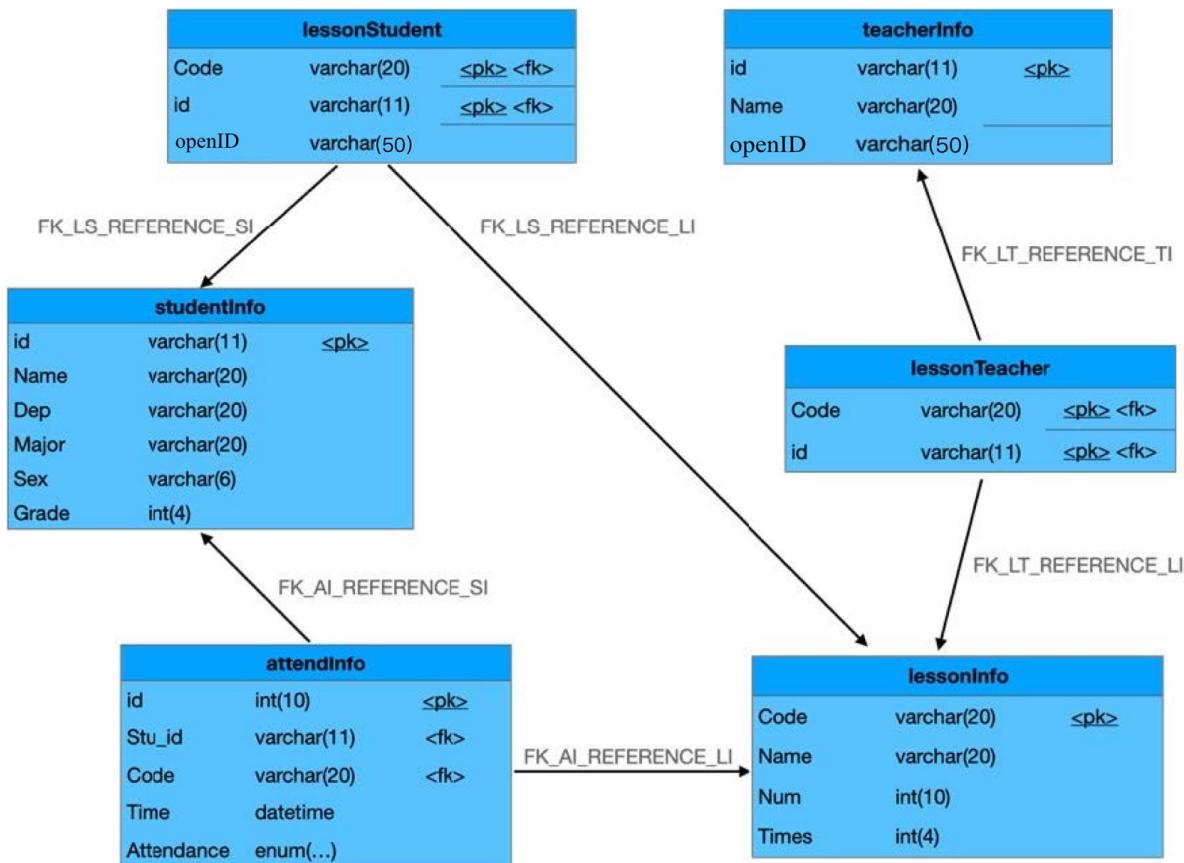


图2-2 数据库模型图

2.5.4 数据库表的建立

根据以上关系模式的建立，我们先在MySQL中创建了checkinsystem数据库。在checkinsystem数据库中建立如下的表。

字段名	字段描述	数据类型	主键	外键	非空	唯一	自增
Id	学号	VARCHAR(11)	是	否	是	是	否
Name	姓名	VARCHAR(20)	否	否	是	否	否
Dep	院系	VARCHAR(20)	否	否	否	否	否
Major	专业	VARCHAR(20)	否	否	否	否	否
Sex	性别	VARCHAR(6)	否	否	否	否	否
Grade	年级	INT(4)	否	否	否	否	否
openID	openID	VARCHAR(50)	否	否	否	是	否

表2-4 studentInfo表

字段名	字段描述	数据类型	主键	外键	非空	唯一	自增
Id	工号	INT(10)	是	否	是	是	否
Name	姓名	VARCHAR(20)	否	否	是	否	否
openID	openID	VARCHAR(50)	否	否	否	是	否

表2-5 teacherInfo表

字段名	字段描述	数据类型	主键	外键	非空	唯一	自增
Code	课程代码	VARCHAR(20)	是	否	是	是	否
Name	课程名	VARCHAR(20)	否	否	是	否	否
Num	课程人数	INT(10)	否	否	否	否	否
Times	出勤统计次数	INT(4)	否	否	是	否	否
Place	上课地点	VARCHAR(20)	否	否	否	否	否

表2-6 lessonInfo表

字段名	字段描述	数据类型	主键	外键	非空	唯一	自增
Code	课程代码	VARCHAR(20)	是	是	是	否	否
Id	老师工号	VARCHAR(11)	是	是	是	否	否

表2-7 lessonTeacher表

字段名	字段描述	数据类型	主键	外键	非空	唯一	自增
Code	课程代码	VARCHAR(20)	是	是	是	否	否
Id	学生学号	VARCHAR(11)	是	是	是	否	否

表2-8 lessonStudent表

字段名	字段描述	数据类型	主键	外键	非空	唯一	自增
Id	记录号	INT(10)	是	否	是	是	是
Stu_id	学号	VARCHAR(11)	否	是	是	否	否
Code	课程代码	VARCHAR(20)	否	是	是	否	否
Time	记录时间	DATETIME	否	否	是	否	否
Attendance	出勤情况	ENUM('缺勤', '出勤', '请假')	否	否	否	否	否

表2-9 attendInfo表

对于以上建立的表，需要有几点说明：

- 1) 在表studentInfo、teacherInfo、lessonInfo中，我们分别对Id/Code和Name两个字段设置了非空约束，这与实际场景相符；只有当学生/老师/课程真实存在的时候，才能进行插入。
- 2) 在表studentInfo和teacherInfo中，我们分别对openID这一段设置了非空约束，由于openID与个人微信号相关，所以这样的设置保证了一个微信号只能登录和绑定一个学生或老师账号，确保了安全性。
- 3) 在表lessonInfo中，我们对Num和Times字段设置了非空约束，并且其默认值均为0，即在插入课程时，默认的选课人数和出勤统计次数为0。
- 4) 在attenInfo表中，我们对Id这一字段设置了自增约束，即记录号在每次插入新的出勤数据时，都会自增。

2.5.5 物理引擎的选择

选择InnoDB引擎。原因是在上述表中存在较多的外键，常用的数据库引擎中只有InnoDB能支持。

2.5.6 索引建立

需要特别对attendInfo表的Stu_id和Code字段建立索引。原因是记录数=学生数×每人平均选课数×每门课平均考勤次数，如在一个学校中有5000名学生，每人平均选课数为5门，每门课的考勤次数为10次，则记录数为25万条。较多的查询都是建立在该表上的，所以需要对其建立索引，以加快查询速度。

```
1 | CREATE INDEX in_code_id ON attendinfo(`code`,stu_id)
```

代码2-3 创建索引

2.5.7 触发器建立

根据需求，在两处地方需要建立触发器，分别为attend表中的Num和Times。在之前创建表中，我们将两字段设置默认值为0。当新的数据插入时，如有同学的选课信息或者考勤信息插入时，需要将其自动进行改变。

针对Num字段，我们建立了一下两个触发器，其实现的功能为：插入学生选课信息时，选课人数加一；删除学生选课信息时，选课人数减一。代码如下所示：

```
1 delimiter $  
2 CREATE TRIGGER tri_testInsert after INSERT on lessonstudent for EACH ROW  
3 BEGIN  
4     DECLARE c INT;  
5     set c = (SELECT num FROM lessoninfo WHERE code=new.code);  
6     UPDATE lessoninfo set num=c+1 where code=new.code;  
7 end $  
8 delimiter ;
```

```
1 delimiter $  
2 CREATE TRIGGER tri_testDelete after DELETE on lessonstudent for EACH ROW  
3 BEGIN  
4     DECLARE c INT;  
5     set c = (SELECT num FROM lessoninfo WHERE code=old.code);  
6     UPDATE lessoninfo set num=c-1 where code=old.code;  
7 end $  
8 delimiter ;
```

代码2-4 创建触发器

针对Times字段，由于该字段进行修改是在老师发布签到任务后，而不是在attendInfo表中插入数据后进行，所以难以通过触发器对其进行修改。我们将这一功能的实现放在了后端的部分，即当老师发布任务后，再对数据库进行修改。

3 功能介绍

3.1 教师端

(1) 首次登录绑定

在教师首次登录该微信小程序后，系统会自动将教师的openID（可以理解为一个匿名的微信号，具有唯一性与不变性，即每个微信在每个应用程序中有唯一一个不变的openID，能够更好地保障人员的信息安全）绑定至该教师的信息中，同时一个微信也仅能绑定一个账号，避免了一人登录多人账号的情况出现。

在这之后，别人无法通过他们的微信登录该教师的账号，该教师也无法绑定他人的账号。

(2) 发起签到任务

在发起签到任务界面中，教师可以选择发起签到的地点与时间，比如老师有事出差，但是又想要进行签到的话，仍可以选择签到地点为上课的教室。在选择完毕后点击确认发起签到，即可以获得一个含有签到时间，地点，以及课程信息的二维码用于学生签到。

(3) 查看签到记录

在历史签到记录界面，教师可以直接查看最近一次的考勤记录信息，以及以前发起签到的全部签到信息。

在缺勤情况查询与修改界面，教师可以选择签到日期，系统会列出在该签到日期中缺勤的人员。

(4) 下载考勤记录

在历史签到记录界面，教师可以通过点击下载考勤记录按钮，获得一张具有每个学生签到记录的考勤记录表。

(5) 缺勤记录修改

在缺勤情况查询与修改界面，教师选择签到日期后，系统会列出在该签到日期中缺勤的人员，教师可以选择其中的人员，将其标记为出勤或请假状态。

(6) 查看统计信息

在统计信息界面，教师可以选择查询的课程（全部课程，或某一门该教师教的课程），以及查询的维度（年级，性别，学院），在选择完毕后，点击查询按钮，系统将会返回该课程不同维度下的出勤，请假及出勤率。

(7) 联系开发人员

在个人信息界面，我们还加入了联系开发者（之一）的按钮，如果用户在使用中有任何的问题，有改进建议，需要进行一些操作等的情况的话，可以通过这一方式联系到我们。

3.2 学生端

(1) 首次登录绑定

在学生首次登录该微信小程序后, 系统会自动将学生的openID该学生的信息中, 同时一个微信也仅能绑定一个账号, 避免了一人登录多人账号的情况出现.

在这之后, 别人无法通过他们的微信登录该学生的账号, 该学生也无法绑定他人的账号.

(2) 进行签到任务

在教师将二维码发布后, 学生可以进行扫码操作, 当扫描了正确的二维码, 且尚在有效签到时间内的話, 才可以进入签到界面开始签到.

在签到界面中, 学生需要点击距离签到地点, 获取与老师规定的签到地点之间的距离, 当这一距离小于200米时, 学生才可以点击签到按钮, 完成签到任务.

(3) 查看签到记录

在历史签到记录界面, 学生可以直接查看自己出勤请假及缺勤的总次数, 以及以前发起签到的全部签到信息.

(4) 查看统计信息

在统计信息界面, 学生可以选择查询的课程(全部课程, 或某一门该学生所上课程), 以及查询的维度(年级, 性别, 学院), 在选择完毕后, 点击查询按钮, 系统将会返回该课程不同维度下的出勤, 请假及出勤率.

(5) 联系开发人员

在个人信息界面, 我们还加入了联系开发者(之一)的按钮, 如果用户在使用中有任何的问题, 有改进建议, 需要进行一些操作等的情况的话, 可以通过这一方式联系到我们.

4 开发平台及框架

Flask

Flask 是一个微型的 Python 开发的 Web 框架, 也被称为“microframework”。Flask没有默认使用的数据库、窗体验证工具。此外,它还有很强的定制性, 用户可以根据自己的需求来添加相应的功能。Flask框架的主要特征是核心构成比较简单, 但具有很强的扩展性和兼容性, 程序员可以使用Python语言快速实现一个网站或Web服务。

——摘自百度百科

在我们小组的设计中,Flask被用于搭建后端对mysql数据库进行管理。

微信开发者工具

微信开发者工具是微信官方提供的针对微信小程序的开发工具, 集中了开发, 调试, 预览, 上传等功能。微信团队发布了微信小程序开发者工具、微信小程序开发文档和微信小程序设计指南, 全新的开发者工具, 集成了开发调试、代码编辑及程序发布等功能, 帮助开发者简单和高效地开发微信小程序。

——摘自<http://zixun.jisuapp.cn/>

在我们小组的设计中，微信开发者工具被用于搭建前端的用户交互界面以及与后端进行交互。

MySQL

MySQL 是一款安全、跨平台、高效的，并与 PHP、Java 等主流编程语言紧密结合的数据库系统。目前 MySQL 被广泛地应用在 Internet 上的中小型网站中。由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，使得很多公司都采用 MySQL 数据库以降低成本。

——摘自<http://c.biancheng.net/view/7668.html>

在我们小组的设计中，MySQL 是我们选用的数据库，用于存储各种表信息以及进行最底层的数据处理。

5 系统安装部署说明

python部署：

python 3:

安装虚拟环境包（已安装可忽略）

```
1 | pip install virtualenv  
2 | pip install virtualenvwrapper
```

创建虚拟环境：

```
1 | cd /... #进入目录  
2 | mkdir wxcheckin #创建项目目录  
3 | virualenv flaskenv #创建虚拟环境 名为flaskenv
```

激活虚拟环境：

在wxcheckin目录下：

```
1 | source flaskenv/bin/activate
```

安装所需库：

```
1 | pip install -r requirement.txt
```

python环境安装完毕。

在backend.py文件中，将第38行，第306行的数据库名、用户名、密码改为本地数据库。

```
1 | # 第38行  
2 | app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+mysqlconnector://用户名:密码  
@localhost:3306/数据库名'  
3 | # 第306行  
4 | engine = create_engine('mysql+mysqlconnector://用户名:密码@localhost:3306/数据库  
名')
```

运行：

```
1 | python backend.py
```

数据库部署：

创建数据库checkinsystem，并source导入checkinsystem.sql文件

小程序：

安装微信开发工具。

新建小程序项目，输入注册号，并选择使用云服务。

我们在此提供一个我们微信小程序的APPID: wx84bcc90a69695251

将小程序文件导入后(只导入其中的“(记得换掉)”文件)，点击创建按钮

如果在本地使用，先运行python文件，再运行微信小程序。

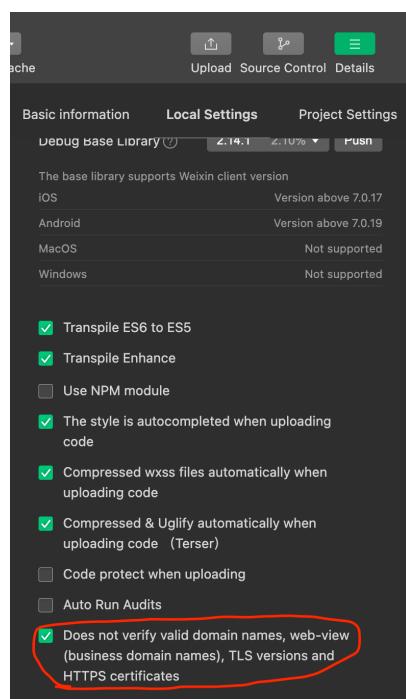
附：系统使用说明

1. 进入小程序

因为该版本尚且处于测试版本，因此并不能像平常的微信小程序那样直接使用，我们提供三种使用方法。

(1)

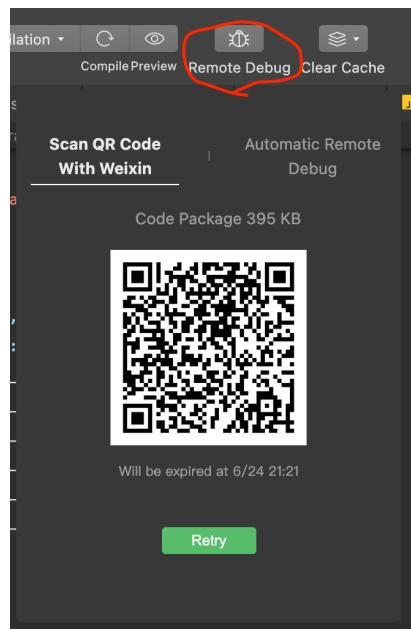
首先启动python文件backend.py。然后在微信开发工具中，请先点击程序右上角详情页面，点击本地设置后，开启不校验合法域名。然后，如使用在本地搭建的后端，请先将app.js文件中，globalData中的path更换为<http://127.0.0.1:5000>。



随后直接在微信小程序开发软件中左侧的模拟器中使用。



(2) 执行与上一种方法一样的操作后，点击真机调试(Remote Debug)，并扫码或自动连接进行使用。



(3) 扫描以下体验版二维码，点击前往体验版页面按钮，进入小程序页面后，点击右上角的三个点，点击右下角进入启用调试功能后，再次点击前往体验版，进入小程序界面出现一个vConsole的标志后才可以使用，如正式发布后则可以直接使用。



扫码点名系统

体验版

开发人员 レオ丶
版本号 1.0.3
提交时间 6-24-2021 12:24:12
描述 レオ丶 at June 24, 2021 12:24 PM
upload

前往体验版

可将此页面分享给其他体验者，或添加收藏



2. 教师登录与相关功能

首先，在开始界面，我们可以根据自己的身份选择“我是老师”或“我是学生”，首先我们先介绍教师端的功能，因此我们首先选择我是老师



然后页面将跳转至个人信息界面,如果不小心点击错误,则还可以在该界面选择“切换身份”,如果没有问题,则可以点击编辑信息登录,只有在登录后才可以使用课程与统计界面,在获取openID成功后,输入正确的姓名与ID后即可以登录.(或者也可以直接点击课程或统计界面的“请先登录”框,点击后也会自动跳转至登录界面)

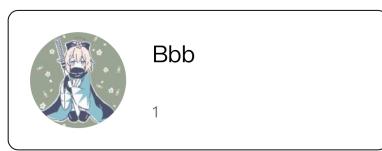


在登录成功后,系统会自动返回我们切换身份的按钮会消失,这是因为我们只允许一个微信号绑定一个人,一个人也只能对应一个微信号,因此登录后不再允许重新登录.

11:55



个人信息



姓名: Bbb
工号: 1

[联系开发者\(之一\)](#)

vConsole



这时我们就可以查看我们的课程了：

11:55



课程信息



vConsole



在选择一门课程之后，跳转到如下界面，老师可以选择发起签到任务，查看历史签到记录，以及进行缺勤情况的查询与修改



vConsole

在历史签到记录界面，教师可以查看最近一次的签到情况，以及全部的历史签到记录，并且可以点击“下载考勤记录”按钮，下载签到记录表(下载成功后自动弹出如图二表格，点击图二右上角的三个点即可进行保存，转发到好友等操作)

11:55

历史签到记录

摸鱼学
课程人数：21
课程代码：BBBS.100001

11:55

0785a15b4602cb533910dc36fc2b...

学号	姓名	2021-04-23	2021-05-07	2021-05-14	2021-06-23	缺勤数	请假数
1708122331	熊燕	√	√	√	√	0	0
1723303651	李洁	√	√	√	√	0	0
17279324051	杜萍	√	√	√	√	0	0
1743958541	杨亮	x	x	x	请假	3	2
17688893201	刘志强	请假	x	请假	x	2	1
1797774981	杨春梅	√	请假	x	请假	0	0
1803462009	魏强	请假	请假	请假	请假	1	0
1809178622	董桂英	√	请假	请假	√	0	0
18641907171	王桂香	x	请假	√	√	1	1
1874571221	张霞	x	请假	√	x	2	0
1894976901	杨丽	√	√	√	√	0	0
1903972701	何健	√	x	√	√	1	1
19656914681	吴建军	请假	x	√	√	1	1
1968786641	齐丹	请假	√	√	√	0	0
1985493131	胡兵	√	√	x	√	1	1
1999790254	吴明	√	√	√	x	1	1
2009808111	赵慧	x	x	x	√	3	0
2019312251	梅海	√	请假	√	√	0	0
2052764901	卢兰英	√	请假	x	√	1	1
2092473351	郑培	√	请假	x	√	1	1
2098952121	陈帅	√	请假	√	√	0	0

最近一次考勤记录

总和	出勤	请假	缺勤
21	16	2	3

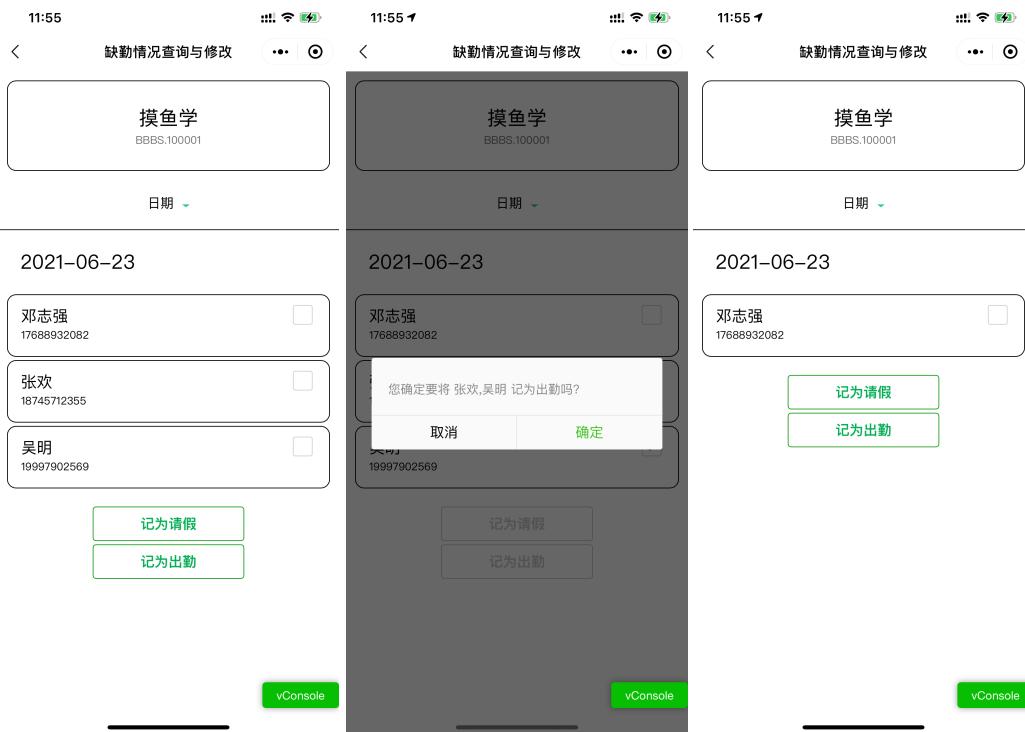
历史记录

日期	出勤	请假	缺勤
2021-04-23	13	4	4
2021-05-07	7	9	5
2021-05-14	12	3	6
2021-06-23	16	2	3

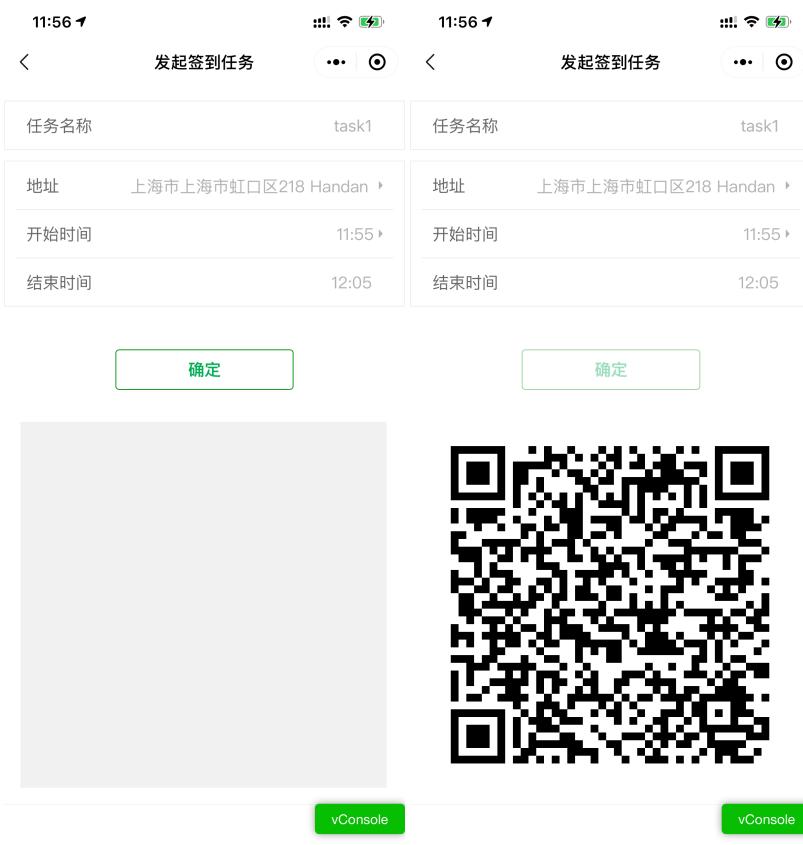
下载考勤记录

vConsole

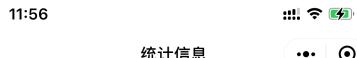
在缺勤情况查询与修改界面，教师可以选择日期，查看缺勤人员，并可以选择学生，将其缺勤状态修改为请假状态或出勤状态，避免了学生上课忘记带手机，或是手机出现故障导致无法签到被记为缺勤的情况发生。



在发起签到任务界面，老师在完整填写任务名称，选择考勤地址，以及选择时间后，点击确定按钮，即可生成用于签到的含有签到时间，签到地点和课程信息的二维码，用于给学生签到。



在统计界面，我们可以通过点击上方的课程与维度文字，选择查看某一门课程或全部课程的 年级，性别或院系维度的出勤情况(只能够查询自己所选择的课程的信息)，点击查询按钮后，就会出现下方的出勤情况框，当某一比例小于5%时，比例则显示为-。



课程 维度

选择的课程: 全部
选择的维度: 年级

查询

出勤情况

(绿色出勤, 黄色请假, 红色缺勤)

2017级

45% 14% 41%

2018级

53% 13% 34%

2019级

45% 17% 38%

2020级

38% 14% 48%

vConsole

课程

统计

我

3. 学生登录与相关功能

介绍完了教师端的功能, 下面将介绍学生端的使用说明

同教师端一样, 我们也需要首先选择“我是学生”按钮, 填写完个人信息后, 才能进一步进行操作

13:25 12:16

登录 个人信息

个人信息

1. 姓名
于淼芃

2. ID
19307130203

登录成功

于淼芃
19307130203
姓名: 于淼芃
学号: 19307130203
学院: 大数据学院
专业: 数据科学
年级: 2019

联系开发者(之一)

vConsole

vConsole

课程

统计

我

首先, 我们也需要选择要进行操作的课程, 在点击后, 将会跳转到扫描二维码页面



在没有签到任务时,我们可以查看自己的历史签到记录,如果您在某一门课上全勤了,我们还会授予您卷王称号

历史签到记录			
总和统计			
总次数	出勤次数	请假次数	缺勤次数
4	3	1	0

历史签到记录			
总和统计			
#	日期	时间	状态
1	2021-03-21	16:10	出勤
2	2021-04-08	15:45	出勤
3	2021-06-08	16:22	请假
4	2021-06-23	16:14	出勤

历史签到记录			
总和统计			
#	日期	时间	状态
1	2021-03-21	15:34	出勤
2	2021-03-28	09:55	出勤
3	2021-04-23	09:58	出勤
4	2021-06-07	10:22	出勤
5	2021-06-23	16:35	出勤

在有签到任务时,我们需要点击扫描二维码的扫描框,进行扫码操作,下面三张图片分别为(1)扫描了超过签到时间的二维码,(2)扫描了不是本课程的二维码或其他错误的二维码,(3)扫描了该课程,且在签到时限内的二维码.当签到按钮为灰色时,无法点击.





在进入签到界面后, 学生需要点击距离签到地点处的小箭头, 系统会自动返回学生距离发起者的距离, 当距离小于200米时, 签到按钮变为可以点击, 否则仍然为灰色, 之后点击签到即可签到成功.



统计界面的设计与教师端的统计界面设计相同, 使用请参考7.2部分。

4. 用户反馈

同时, 如果有什么特殊的情况, 比如需要对小程序的建设有什么建议, 或者是想要来聊天之类的, 还可以点击个人信息界面的“联系开发者(之一)”按钮, 我们将会解答您的疑惑

12:16

12:16

个人信息

... | ○



于淼芃

19307130203

姓名: 于淼芃
学号: 19307130203
学院: 大数据学院
专业: 数据科学
年级: 2019

联系开发者(之一)



vConsole

